SPECIFICATION

Electronic Version 1.2.8

Stylesheet Version 1.0

METHOD OF MODELING PRODUCT DEMAND SUBJECT TO A LARGE NUMBER OF INTERACTIONS

Background of Invention

[0001] *FIELD OF THE INVENTION* This invention relates in general to methods and data processing system readable media, and more particularly, to methods of modeling operating parameters and data processing system readable media having software code for carrying out those methods.

[0002] *DESCRIPTION OF THE RELATED ART* Selecting prices for thousands of items is a difficult proposition. The current practice indicates that retailers may be over-discounting products, with as many as 25%-30% of items being sold at some price discount. With profit margins so low already (approximately 1.5% is some business sectors), many retail stores may not be able to sustain aggressive price discounting.

[0003] Pricing is made difficult by the fact that products interact with each other. Decreasing the price of one juice item to increase traffic may merely result in the cannibalization of a more profitable juice brand (as consumers switch from one brand to the other), without increasing demand for all juice products. Similarly, raising prices may have pronounced consequences across category boundaries, such as decreasing the number of items bought in distant or unrelated categories due to a general reduction in store traffic.

[0004] Retailers have long been aware of price-demand interactions and have developed various strategies for coping with these effects. One common strategy is known as

"loss leader pricing." "Loss leaders" are products that are kept at discounted prices because they are known to be high profile, common, and easily comparable between retailers. Typical loss leaders for grocers include milk, bread, eggs, and juice. Loss leaders are presently determined by retailer experience. A problem is correctly identifying which, if any, products are to be loss leaders.

[0005] Automated analysis of product interactions has as yet been limited to very small numbers of products, often within the same category. Working with all product interactions is often practically infeasible because of the number of potential interactions that need to be examined. Assuming 100,000 items are examined, there could be 10 billion interactions.

[0006] A need exists for a comprehensive approach to modeling retail operations which incorporates knowledge of product interactions, consumer demand, storewide effects and the like. Furthermore, since most retail stores have thousands of items and millions of transactions, a need exists to deal with product interactions in a comprehensive, automated, computationally fast and efficient manner.

Summary of Invention

[0007] A method can be implemented on a data processing system to model an operating parameter. The method may incorporate knowledge of purchasing interactions between items in the inventory. The method may also incorporate knowledge of externalities (external events) that may affect the sales of products. The method may use a matrix of weighing factors. All weighing factors may be zero except for items where quantities sold of a particular item are significantly affected by a price change with that item or another item. Use of the method can reduce the computational time and resources required to build the model and allows the method to operate on a large amount of data in a reasonable amount of time. Applications of the method can include but are not limited to demand forecasting, price optimization, what-if analyses, promotion planning, and inventory control.

[0008]

In one set of embodiments, a method can be used to model an operating parameter for a vendor. The method can comprise determining an effect of a variable

on quantities of items sold by a vendor to determine which of the items are significantly affected by the variable. The method can further comprise generating a matrix that includes weighing factors. For each item that is more significantly affected by the variable, assigning a non-zero value to its corresponding first weighing factor. For all other items that are less significantly affected by the variable, assigning values of zero to their corresponding first weighing factors. The method still further comprises calculating the operating parameter using the matrix.

[0009] In other embodiments, a data processing system readable medium can have code embodied within it. The code can include instructions executable by a data processing system. The instructions may be configured to cause the data processing system to perform the methods described herein.

[0010] The foregoing general description and the following detailed description are exemplary and explanatory only and not restrictive of the invention, as claimed.

Brief Description of Drawings

[0011] The present invention is illustrated by way of example and not limitation in the accompanying figures, in which like references indicate the same elements, and in which:FIG. 1 includes an illustration of a functional block diagram of a system used with data mining; FIG. 2 includes an illustration of a data processing system storage medium including software code having instructions in accordance with an embodiment of the present invention;FIG. 3 includes a process flow diagram for determining an operating parameter for a vendor; andFIGs. 4 and 5 include plots of the 100 top negative correlations and 100 top positive correlations, respectively for a price change in a specific item.

[0012] Skilled artisans appreciate that elements in the figures are illustrated for simplicity and clarity and have not necessarily been drawn to scale. For example, the dimensions of some of the elements in the figures may be exaggerated relative to other elements to help to improve understanding of embodiments of the present invention.

Detailed Description

[0013] A method can be used to model an operating parameter for a vendor. Fast and

computationally efficient significance tests can be first performed between the observed prices of items over time, externalities (as independent variables), and a target item's demand over time (as the dependent variable). After determining which variables appear to have a significant impact on item demand, weighing factors for those significant variables are then determined using stepwise regression. The weighing factors for all other variables can be assigned a value of zero. In one set of embodiments, the number of weighing factors having a non-zero value within the matrix (or matrices) may be no more than approximately ten percent of all weighing factors within the matrix (or matrices), and often may be no more than approximately one percent. By limiting the number of non-zero weighing factors, the time needed to generate all the weighing factors for a matrix (or matrices) is reduced. Also, the operating parameter can be calculated more quickly because of the relatively lesser number of weighing factors having a non-zero value. A data processing system readable medium can have code including instructions for carrying out the method. The present invention is defined by the appended claims are better understood after reading the descriptions of embodiments below.

[0014] FIG. 1 includes a system 10 for mining databases. In the particular architecture shown, the system 10 can include one or more data processing systems, such as a client computer 12 and a server computer 14. The server computer 14 may be a Unix computer, an OS/2 server, a Windows NT server, or the like. The server computer 14 may own a database system, such as DB2 or ORACLE, or it may have data on files on some data processing system readable storage medium, such as disk or tape.

[0015] As shown, the server computer 14 includes a mining kernel 16 that may be executed by a processor (not shown) within the server computer 14 as a series of computer-executable instructions. These instructions may reside, for example, in the random access memory (RAM) of the server computer 14. The RAM is an example of a data processing system readable medium that may have code embodied within it. The code can include instructions executable by a data processing system (e.g., client computer 12 or server computer 14), wherein the instructions are configured to cause the data processing system to perform a method. A focus of the methods may be to predict operating parameters for a vendor's store. The method is described in more

detail later in this specification.

[0016] FIG. 1 shows that, through appropriate data access programs and utilities 18, the mining kernel 16 can access one or more databases 20 or flat files (e.g., text files) 22 that contain data chronicling transactions. After executing the instructions for methods, which are more fully described below, the mining kernel 16 can output relevant data it discovers to a mining results repository 24, which can be accessed by the client computer 12.

[0017] Additionally, FIG. 1 shows that the client computer 12 can include a mining kernel interface 26 which, like the mining kernel 16, may be implemented in suitable software code. Among other things, the interface 26 may function as an input mechanism for establishing certain variables, including the number of significant terms to use, window-size for timeseries smoothing, and so on. Further, the client computer 12 may include an output module 28 for outputting/displaying the mining results on a graphic display 30, a print mechanism 32, or a data processing system readable storage medium 34.

[0018] In addition to RAM, the instructions in an embodiment of the present invention may be contained on a data storage device with a different data processing system readable storage medium, such as a floppy diskette. FIG. 2 illustrates a combination of software code elements 204, 206, 208 and 210 that are embodied within a data processing system readable medium 202 on a floppy diskette 200. Alternatively, the instructions may be stored as software code elements on a DASD array, magnetic tape, conventional hard disk drive, electronic read-only memory, optical storage device, CD ROM or other appropriate data processing system readable medium or storage device.

[0019] In an illustrative embodiment of the invention, the computer-executable instructions may be lines of compiled C^{++}, Java, or other language code. Other architectures may be used. For example, the functions of the client computer 12 may be incorporated into the server computer 14, and vice versa. FIG. 3 includes an illustration, in the form of a flow chart, of the operation of such a software program.

[0020] Communications between the client computer 12 and the server computer 14 can be accomplished using electronic or optical signals. When a user (human) is at the client computer 12, the client computer 12 may convert the signals to a human understandable form when sending a communication to the user and may convert input from a human to appropriate electronic or optical signals to be used by the client computer 12 or the server computer 14.

[0021] FIG. 3 includes a flow diagram for a method of determining an operating parameter for a vendor. The method can comprise determining which price changes have a significant effect on demand for items carried by the vendor (block 344) and determining which other variables have a significant effect on demand for items by the vendor (block 346). The method can also comprise generating matrix of weighing factors for variable-quantity interactions (block 362). The method can further comprise calculating the operating parameter using the matrix (block 364). Details of the method are given in the subsequent paragraphs that follow.

[0022] 1. Collect and group the data

[0023] The transaction data can be collected from Point of Sale (POS) records or another source (such as general ledgers), which might have details of customer or vendor orders (e.g., the method may use warehouse shipment data since this also shows quantities of items and the approximate dates of those sales). The different items can be denoted by a Universal Product Code (UPC), a Stock Keeping Unit (SKU), or any other convenient identifying code.

[0024] Transaction data is often stored in large databases, such as database 20, with records essentially unordered. If item timeseries are constructed from such data directly, all item timeseries may need to be maintained in memory. As each transaction is encountered, it would be added to the timeseries for whichever item to which that data referred. Such an approach is practically infeasible for large amounts of data and large numbers of items. Approximately 2.4 gigabytes of RAM memory space may be used for a million different items and a timeseries of 600 days. Many computers may need additional hardware to process this much data.

[0025] A technique can be used for efficiently building the timeseries data for price and quantity for each item may use relatively constant-sized memory and be performed in a time which is a linear function of the number of rows. The method involves changing the row order of the transaction data so that all records for each item are in contiguous rows (note the data does not need to be sorted). This will be referred to as a "grouping algorithm" (note that this is different from the SQL "group by" command). Grouping algorithms can run in linear time, meaning the time to perform the task is directly proportional to the amount of input data, and use disk resources for "scratch pad" storage.

[0026] Sorting algorithms such as "merge sort" can also be used to contiguously re-order rows, so that all item records are together. Merge-sorts can be performed using disk storage with a time complexity of $O(\Phi * \log(\Phi))$, where Φ is the size of the input data.

[0027] Sorting algorithms may not be as fast as a grouping algorithm; however, both approaches (sorting or grouping) are good for handling very large amounts of data with the limited RAM available to current digital computers 12 or 14. As described in the next section, the re-ordering can be used to construct all timeseries in memory which does not grow with the number of items, or number of rows. Thus, the grouping or merge-sort algorithm can run using a constant amount of memory and be performed in a time that is $O(\Phi)$ or $O(\Phi * \log(\Phi))$.

[0028] 2. Build quantity and price timeseries for each item.

[0029] After the data is re-ordered by item, timeseries for each item's price and quantity can be built. The timeseries can be built by scanning the item-ordered transaction data one row at a time. Information gleaned from the transaction data may include the item, unit price of the item and the number of units sold (quantity). When all transaction records for an item have been accessed, there can be no further instances of that item record in later rows of the transaction data since the contiguous reordering operation described above has been performed. The data processing system can "package up" the timeseries for that item and "flush" the timeseries information from RAM, which frees up RAM space.

[0030] During "packaging," the timeseries for the item is completed and can be sent to a database or file containing the timeseries data. After packaging, the data processing system can free the space within RAM occupied by the last item's timeseries data before processing timeseries information related to the next item. Thus, an advantage to the procedure is that item timeseries can be computed using an amount of memory which does not grow with the number of items.

[0031] There may be other considerations when building these timeseries. At a retail store, many prices may be recorded throughout a single day since customer prices are often affected by the use of store coupons, manufacturer coupons, and so on. Because of this, the price timeseries may be set to equal the mean or the median price of an item on a particular day, and the quantity timeseries can be the sum of units sold on that day.

[0032] Days on which no transaction for a particular item occurred also present difficulties for timeseries creation. On such days, there is no record of the price at which an item was ticketed. The price timeseries will erroneously have "zeros" on those days, even though the item was still on sale.

[0033] "Price filling" may help correct these price timeseries errors. Price filling involves finding the last day on which the item was sold, and then filling each subsequent day forward in time, when no transaction occurred, with that last observed price. Price filling may not be completely accurate because it may not reflect when a price change occurred when were no sales for the item occurred on that date, and it may not reflect days the store was closed (e.g., holidays).

[0034] Alternatively, a calendar with store prices and dates can be used to resolve what price items were on particular days during which no sales were recorded. Other methods may also be possible.

[0035] Moving sum windows may be used to help address the continuity of the quantity timeseries. Many products in a store will tend to be "slow moving" and may sell only a few units each week, with several days going by between sales. Such products are difficult for conventional techniques such as ARIMA to forecast, since they are a

discontinuous function. To address this problem, it is possible to use moving sums to replace every day in the timeseries with the sum of that day and each of the X (e.g., 30) days that follow. This technique can improve predictive accuracy because the low-frequency patterns (seen more with items that only occasionally are sold) may be easier to predict than the high frequency patterns. The moving average exaggerates those lower frequency patterns. The 30-day windows are also intuitively appealing because forecasts can be interpreted as "W units per 30 days". Windows other than 30 days in width may also be used, for instance, slower-moving merchandise may require longer time windows.

[0036] 3. Perform fast significance tests for price-quantity correlations.

[0037] After historical 30-day rolling-sum quantity and price timeseries for each item are obtained, predictive models can be built to predict the demand of each item given the prices of other items.

[0038] Using all of the cross-price variables would be difficult due to their sheer number. In a store with approximately 40,000 different items, for each item, there could be up to 39,999 items, plus its own price, that may be affecting its sales. Under normal circumstances, to derive an equation for one item, 40,000 regression equations would need to be tested in a stepwise procedure, with the stepwise procedure requiring perhaps 20-30 iterations before converging on a solution. Thus, performing stepwise regression on all items might require 32 billion regression computations.

[0039] Instead of that method, an efficient significance test may be performed to determine which terms (price change or other variables) are likely to have a significant effect on demand for items carried by the vendor (blocks 344 and 346 in FIG. 3).

[0040] In one embodiment, a correlation matrix *Corr* for each pair of items may be created. Each correlation factor ($Corr_{ij}$) represents an interaction between the demand series of a target item and the price series for each of the approximately 40,000 items. The *(i,j)* th element of this matrix can be determined by the following equation.

[0041]

$$Corr_{ij} = \frac{\sum_{t=1}^{N} \left((price_{ti} - E[price_i]) \cdot (qty_{tj} - E[qty_j]) \right)}{\sqrt{\sum_{t=1}^{N} \left((price_{ti} - E[price_i])^2 \cdot (qty_{tj} - E[qty_j])^2 \right)}}$$

[0042] where,

[0043] $price_{ti}$ is the price of item i at time t;

[0044] $E[price_i]$ is the expected price of item i;

[0045] Qty_{ij} is the quantity of item j sold with item i is at price $_{ti}$; and

[0046] $E[qty_i]$ is the expected quantity for item j.

[0047] The expected price for item i and the expected quantity for item j may be the average price for item i and the average quantity for item j based on data collected during the time span (t from 1 to N) or recent historical data. Basically, the correlation factor can be representative of how a change in quantity of item j is affected by a price change in item i. Each correlation factor may have a value between -1.0 to +1.0.

[0048] Other tests besides correlation can be used to determine significance, including Spearman's rank correlation, measures involving entropy, and so on. However, correlation is recommended because it can be computed quickly and with little memory on large amounts of data. Only the top N positive and negative correlation factors may be selected. A space-efficient algorithm is described below to find the N most significantly related items. This algorithm can run on a computer in a time that scales with $O(I^2 * R)$ and memory that scales with $O(I*(3+N))$, where I is the number of items, R is the number of days in the timeseries and N are the number of most positively and negatively correlation factors.

[0049] Before beginning to calculate correlations, the mean of each variable is determined in memory. The amount of memory used scales O(I). In one embodiment, a row-by-row scan of the data for each variable is performed. For each variable an accumulator for sumX and a counter for number of rows R encountered up to that point are kept in memory. Each time a new value of X is encountered, that value is added to sumX, and N is incremented by 1. After the scan is completed, sumX is

divided by N to give the mean.

[0050] After means have been computed, client computer 12 of the server computer 14 can determine the N items (per dependent item) that are most strongly correlated. For each item X that could have an effect on item Y, a correlation factor is calculated using the memory-efficient method below.

[0051] Accumulators for "sum of x times y" (sumXY), "sum of x times x" (sumXX) and "sum of y times y" (sumYY) are used and may reside in memory. Each time a datapoint is encountered, the mean is subtracted, and the accumulators are updated with the result. After completing a scan of all the data (in this case a timeseries that might consist of 600 points or so), the final correlation factor between X and Y, $Corr_{X,Y}$ is calculated by taking sumXY/sqrt(sumXX * sumYY).

[0052] If the correlation factor is positive, a variable called "bottom-significant-pos-correlation" can be compared to the correlation factor. If the correlation factor has a value further from zero compared to the bottom-significant-pos-correlation, then item X can be inserted as a correlation factor into the list of "significant-pos-correlations." If there are already N items in the list, the bottom significant variable is deleted, and the method is used to replace bottom-significant-pos-correlation with the next lowest correlation factor within the list. A similar procedure is performed for negative correlation factors.

[0053] FIGs. 4 and 5 illustrate the top 100 negative and positive correlations, respectively, for a specific target item. As can be seen in these figures, a relatively small number of items have a strong correlation with a price change of the target item (item i). The strength of those correlations drop off rapidly. Thus, although there are over 40,000 items in this example, only a relatively small number of them impact the demand of the target item and should be incorporated into a stepwise variable pool.

[0054] FIGs. 4 and 5 show that many items (perhaps 70%) might undergo no price change in an entire year. Thus, the demand changes for these items are not driven by their own price at all, a fact which contradicts most intuitions about selling products! Instead, their demand is most likely being driven by external factors, and particularly

the interactions of other products carried by the vendor. For example, customers may see low prices for lettuce and canned soup and then buy many more products at the vendor's store.

[0055] In many instances, the price of an item is not one of the most highly correlated drivers affecting its quantity (demand). Many instances may occur in which the correlation factor for the target item was not one in the top one percent of most negatively or most positively correlated items when the price of the target item is changed.

[0056] The effects between the same two items may be different. For example, if a first item is a package of hot dog buns and a second item is package of frankfurters, a price change in frankfurters may have a greater effect on the demand for hot dog buns sold, whereas, a price change in hot dog buns may have a lesser effect on demand for frankfurters. Therefore, the effects do not have to be symmetric.

[0057] The cross effects may extend beyond one category. Hot dog buns and frankfurters are in different categories. Hot dog buns are a bread-related product, and frankfurters are a meat-related product. Therefore, the method can be used to examine not only the items within a same category but also items within different categories.

[0058] Many items may be weakly correlated, and those interactions can be largely ignored. The price of the dishwashing liquid may have a relatively insignificant impact on the demand for a can of soup, and likewise, the price of a can of soup probably may have very little impact on the demand of dishwashing soap. Therefore, the relationship between dishwashing liquid and a can of chicken soup may be significantly smaller as compared to the effects of hot dog buns and frankfurters. Weak correlations are not required are zero-ed out, and do not contribute to the model. The top positive and negative correlations (values farthest from zero) for a price change can be selected.

[0059]

The number of positive and negative correlations can be chosen by the user. The number can be less than approximately ten percent of all items, and often may be less than approximately one percent. In a store carrying about 40,000 different items, the

top 50 positive and top 50 negative correlated items can be used. Using more than this number can increase computation time and frequently does not significantly improve accuracy. In an alternate embodiment, less than 100 correlated items (i.e., top 50 positive and top 50 negative correlated items) may be used.

[0060] 4. Perform fast significance tests for correlations between other variables and quantities of each item sold.

[0061] The method further includes determining which "other variables" have a significant effect on demand (quantities) for items carried by a vendor (block 346). These "other variables" may include externalities, lag-demand-terms, and global-price-terms, or the like.

[0062] Externalities may include variables for days of the week (Saturday, Sunday, etc.), months of the year, seasons, and the presence of holidays for which the forecast is being generated. For example, the sales of cocoa and hot chocolate may change with the seasons, or the sales of alcohol may change based on the day of the week.

[0063] Lag-demand terms are the quantity of an item sold, some days in the past. For instance, the lag-demand of apples with lag=5 is the quantity of apples sold 5 days ago. The lag-demand terms can help capture recurrent patterns in product consumption. For example, newspaper sales tend to be the same each week (note that in this example, externality variables for day-of-week might accomplish the same thing).

[0064] Global price terms are aggregated summaries of the prices on different days. For example, an average-price variable for a given day can be used to indicate when the vendor has significantly decreased prices of many items at the vendor's store.

[0065] The feature in common with all of the above variables is that they are usually available from the transactional data that a retailer collects in the course of its normal business activities. Because of this, the method can be made to work without any external third party sources of information (which may have complications related to for consumer privacy), or without elaborate records of promotion or newspaper advertising events. Still, such variables could be incorporated if those histories were

available. For many retailers, transaction data (such as point-of-sales data) may be the only easily accessible source of information. The method can be used to work with only this source of information; however, the method may be adapted to include other sources of information if they become available.

[0066] In attempting to identify which of these factors affect the demand of an item j, the same significance test procedure used for price-terms can be applied to these other variables. The top positive and negatively correlation factors are saved and incorporated into a stepwise regression model that is described later.

[0067] At this point, the method has been used to determine the effects of variables on quantities of the items sold within the vendor's store. The method can be used to determine which of the items are more significantly affected by the variable compared to the other items.

[0068] 5. Generate weighing factors for variables and incorporate into model.

[0069] After price changes and variables having a significant impact on demand have been identified, the method can be used to generate a matrix of weighing factors for variable-quantity interactions (block 362). A stepwise regression can be performed using only the top positive and negative correlated items (instead of all items) to obtain the more significant weighing factors. A stepwise regression for weakly correlated items does not need to be performed because the weighing factors for those items may be assigned a value of zero.

[0070] Weighing factors are determined by stepwise regression and are placed into the matrix of weighting factors for the varible-quantity interactions. All other elements (less significantly affected or weakly correlated) in the matrix may be assigned a value of zero. Typically, no more than approximately ten percent of all elements within the matrix may have a non-zero value, and often, no more than approximately one percent of all elements has a non-zero value.

[0071] The method can be used in calculating an operating parameter using the matrix (block 364). The following includes an approximation that can be used with the model.

[0072] Q is approximately equal to P x W + E x B + C

[0073] where,

[0074] Q can be a vector representing quantities of items

[0075] [Q1, Q2, Q3, ... ,Qn];

[0076] P can be a vector representing prices of those items

[0077] [P1, P2, P3, ... ,Pn];

[0078] W can be a square matrix representing weighing factors for price-quantity interactions

[0079]

$$\begin{bmatrix} W1\,1, W1\,2, W1.3, .., W1\,n \\ W2\,1, W2.2, W2.3, . , W2\,n \\ W3.1, W3.2, W3.3, ..., W3\,n \\ . \\ . \\ Wm1, Wm2, Wm3, . , Wmn \end{bmatrix};$$

[0080] E can be a vector representing other significant variables including externalities, lag-demand and global-price

[0081] [E1, E2, E3, ... ,En];

[0082] B can be a matrix of the weights representing weighing factors for other significant variable-quantity interactions

[0083]

$$\begin{bmatrix} B1.1, B1.2, B1.3, . , B1.n \\ B2.1, B2.2, B2.3, ..., B2.n \\ B3.1, B3.2, B3.3, ..., B3.n \\ . \\ . \\ . \\ Bv.1, Bv.2, Bv.3, . , Bvn \end{bmatrix}; \text{and}$$

[0084] C can be a vector representing constants for those items (determined empirically using linear regression)

[0085] [C1, C2, C3, ... ,Cn].

[0086] Given the above model of demand Q, we can also predict revenue R and profit Z for each item:

[0087] $R = Q * diag(P)$

[0088] $Z = R - Q * diag(C)$

[0089] The total quantity, revenue, or profit generated on any given day can be obtained by summing either Q, R, or Z, respectively.

[0090] The construction of W can provide insights into the various product interactions inside the store, as will be described later. For the index numbers, first number designates the price of an item and the second number designates the item whose quantity may be affected by the price change in a first item. A diagonal of the matrix going from W1.1 to Wn.n reflects how demand of items are affected by their own prices.

[0091] The effects between the same two items may be different. W1.2 can reflect the change in demand for item #2 given a price for item #1. Likewise, W2.1 reflects the change in demand for item #1 given a price for item #2. Note that W1.2 and W2.1 are not necessarily equal and, in most cases, are not equal. For example, if item #1 may be a package of hot dog buns and item 2 may be a package of frankfurters.

[0092] The matrix of weights for the other variables-quantity interactions may have rows corresponding to each of these other variable terms and columns corresponding to quantities of items sold corresponding to that variable. Similar to matrix of weights for price-quantity interactions, most of the elements in matrix for the other variable-quantity interactions may be zero except for those items exhibiting the most significant changes (most strongly correlated). The number of elements having a non-zero value within each row of a matrix of weighing factors for the other variable-quantity interactions can be similar to the number of elements having a non-zero

value within each row of a matrix of weighing factors for the price-quantity interactions (ten percent, one percent, fifty items, five items, or the like).

[0093] Note that the two matrices could be combined into one matrix. The "P x W + E x B" can be replaced by "F x G," where F is a vector and G is a matrix.

[0094] The approximation can be varied, if desired. In one embodiment, the approximation can be an equation. In another embodiment, only a single price change on a specific day may be examined. In this instance, the price vector can be replaced by a single price, the weighing factor square matrix can be a row vector, the externalities/lag-demand/global-price (E) can be replaced by a single value, the weights for externality-quantity interactions (B) can be replaced by a row vector, and the constants vector is replaced by a single value. The approximation can be performed relatively quickly because virtually all terms can be first order terms or be zeroed out.

[0095] In the approximation, more or fewer terms could be used. For example, some or all terms in E may be omitted, however, accuracy may be affected. If omitting E terms does not significantly affect the approximation, the model can be simplified and take less time to calculate. The calculation for the operating parameter can be used in many different applications, some of which are described below.

[0096] 6. Applications

[0097] A. Demand forecasting.

[0098] Demand forecasting can be performed for individual items, or for overall sales at the store. Demand forecasting can be achieved by inserting the prices and externalities for a particular day, and then calculating the value of Q for that day. A manager of a store may know what items he or she will put on sale a few weeks from now. The method can be used to determine the quantities of items that will need to be acquired before the sale to reduce the likelihood of running out of the items on sale or their complementary items (high positive weighing factors) or having excessive inventories for other items (high negative weighing factors).

[0099] B. Price Optimization.

[0100] Another implementation may be for price optimization. This involves "solving for P," the price for each item, in order to optimize quantity, revenue, or profit. Because the model "incorporates" the interactions between many products, the model can be used to discover that the price of one item should be dropped because that will increase sales of many items.

[0101] A computational advantage of the model for price optimization is that because of the matrix formulation, derivatives for profit with respect to price can be analytically solved. For example, let P, W, E, B and C be as defined previously, and let c equal a vector of margins ("costs") for each item. The vector of first derivatives of revenue with respect to price are:

[0102]

$$\frac{drev}{dprice_i} = P \cdot W^T + P \cdot W + E \cdot B + c$$

[0103] where,T indicates the transpose of the matrix.

[0104] Derivatives for profit with respect to price are:

[0105]

$$\frac{dprofit}{dprice_i} = P \cdot W^T + P \cdot W + E \cdot B + c - W \cdot diag(C)$$

[0106] The second derivatives of both revenue and profit with respect to price are:

[0107]

$$\frac{d^2 profit}{dprice_i^2} = W^T + W$$

[0108] Because these derivatives are available, fast gradient descent optimization methods can be employed for maximizing profit. For example, a simple gradient ascent procedure for finding optimum prices is to repeatedly set:

[0109]

$$P = P + \alpha\left(\frac{dprofit}{dprice}\right)$$

[0110] where $1 > \alpha > 0$, until the change in P is less than a certain tolerance. Using the second derivatives, Newton"s optimization method will converge on a profit maximum by repeatedly applying the following formula:

[0111]

$$P = P + \left[\left(\frac{d^2 profit}{dprice^2}\right)^{-1} \cdot \left(\frac{dprofit}{dprice}\right)^{T}\right]^{T}$$

[0112] The process is stopped when the change in P is below a certain tolerance. Practitioners in the field of optimization often employ variants of these methods, such as conjugate gradient search, momentum, adaptive parameters, or non-gradient methods may be used.

[0113] C. What-if Analysis.

[0114] A powerful implementation of the method can be its use for what-if analyses. The user can interactively set prices (alter some of the values in P) and observe the outcome on the store in terms of quantity, revenue, or profit. For instance, the user could see that after increasing some prices, demand (traffic) may drop but profit will increase. The user might then decide that the drop in traffic is unacceptable, and he or she will not implement that price change. Alternatively, the manager of the store may be planning a future promotion. The user can enter those prices and predict what will happen. The model is designed with a general purpose computer in mind because the computer can look at more interactions in a given time frame than reasonably possible by humans. Still, there is no theoretical reason why the method cannot be performed by a human.

[0115] D. Inventory Control.

[0116] Additionally, capacity planning for the store and inventory management may be possible. The model may be used to predict the sales of products weeks in advance. The manager of the store may order additional inventory in advance. The model may also be used to "spot" future dates at which inventory may run below safety stock levels. An alert can be automatically generated and sent to managers when additional inventory should be ordered for those dates. Similarly, the model may be used to

reduce the likelihood of carrying too much inventory, which is inefficient and robs space for other items which need to be kept on-hand.

[0117] E. Parasite-host analysis

[0118] In another application, an analysis of each item i can be performed to determine which products are cannibalizers, which products are parasitic on it, which products are symbiotic with it, and which are suckers. Cannibalizers can be found by looking for positive weighing terms from the price of items j (the cannibalizer) onto item i (the victim). Both products cannibalize if W_{ij} and W_{ji} are both positive. Parasites are items for which a price drop in item j causes a decrease in demand for item i, but for which a price drop in item i does not decrease demand for item j. Symbiotic items are items where both W_{ij} and W_{ji} are negative. Thus, a price drop in item i causes increase in demand for item j and vice versa. This kind of analysis can be useful for manufacturers since they can determine which products are undercutting their sales.

[0119] F. Other Applications.

[0120] In still another application, the method may be used for specific situations. For example, a store no longer wants to carry a particular item. What is a good way to deplete existing inventories? One way may be to put it on sale, but if the product has a low margin, the sale may result in a loss. The store may find it more efficient to drop the prices of its driver items (high positive weighing factors). Alternatively, the store may raise the prices of other selected items (high negative weighing factors).

[0121] Implementation of the methods described herein may significantly improve profitability. For price optimization (to improve profitability), profit margins may increase by approximately five percent. This improvement may be the difference between a store losing money and making a profit.

[0122] The embodiments of the present invention can be implemented in software to be used on the client computer 12 or the server computer 14. The grouping or sorting of the data may reduce the likelihood that additional hardware (e.g., more RAM) is needed. Most of the data analysis is related to determining correlations and the weighing factors for the matrices. Using weighing factors speeds calculating, is more

efficient, and still gives reasonably good accuracy for results.

[0123] The data collected can be all internal to the store. By internal, it is meant that the data is collected through the normal events within the store itself. For example, the sales receipts can be used. Information, such as identities of customers, their ages, any of their demographic information, geographic locations of the store, and the like, are not needed for the calculation. Anonymous transaction data (without customer-identifying codes) can be used.

[0124] The method can become more valuable as the number of different items carried by a vendor increases. The vendor may use the method on any portion or all of its operation. For example, the vendor may use the method when modeling one store or a chain of stores. Besides grocery stores, the embodiments can also be used for hardware stores, department stores, specialty stores, or nearly any store.

[0125] In the foregoing specification, the invention has been described with reference to specific embodiments. However, one of ordinary skill in the art appreciates that various modifications and changes can be made without departing from the scope of the present invention as set forth in the claims below. Accordingly, the specification and figures are to be regarded in an illustrative rather than a restrictive sense, and all such modifications are intended to be included within the scope of present invention.

[0126] Benefits, other advantages, and solutions to problems have been described above with regard to specific embodiments. However, the benefits, advantages, solutions to problems, and any element(s) that may cause any benefit, advantage, or solution to occur or become more pronounced are not to be construed as a critical, required, or essential feature or element of any or all the claims. As used herein, the terms "comprises," "comprising," or any other variation thereof, are intended to cover a non-exclusive inclusion, such that a process, method, article, or apparatus that comprises a list of elements does not include only those elements but may include other elements not expressly listed or inherent to such process, method, article, or apparatus.